Your Spectrum

Časopis pravého Spectristy

YS #11: listopad '99

Zlincon 2000
Colony
The Sentinel
Star Raiders II
Spectrum 256!?

YOUR SPECTRUM 11/99
časopis určený výhradně pro uživatele počítačů ZX Spectrum a kompatibilních

Distribuce, předplatné:
8BitCompany Publishing
Tomáš Modroczki
Pražská 2532
438 01 Žatec
Česká republika
tel.: 0602/472579

Adresa redakce:
8BitCompany
Martin Blažek
Luční 4570
760 05 Zlín
Česká republika
tel.:0603/543256

Internet: www.8bc.com
e-mail: 8bc@8bc.com

Redakční rada:
Martin Blažek-Blažko/systems **-BLS-**
Jan Kučera-Last Monster **-LMN-**
Tomáš Modroczki-A. I. D. S. **-AIDS-**
Rudolf Kozel **-STRY-**

Obsah YS 11/99:

Toto číslo je věnováno Tritolovi, který v těchto dnech dokončuje první pořádný file manager pro MB-02+ na ZX Spectrum. Je to opravdový kolos.





Vážení Spectristé,
Jak se máte? Po delší době jsem se opět dostal k tomu, abych vás oslovil na úvodní stránce časopisu YS. Spousta okolních povinností mě přinutila, abych část redakční práce na YS přenesl na své kolegy, které tímto zdravím a kterým děkuji za skvělou práci.
I v tomto čísle YS vám přinášíme recenze pár klasických spectráckých her, několik článků o programování a programovacích jazycích. Také otevíráme zbrusu nový seriál, který se velmi důkladně zaměřuje na zpracování grafiky na ZX Spectru. Autorem je (jak se ostatně dalo čekat) legenda spectrácké scény Slavo Labský (BUSY). O revolučním grafickém zpracování grafiky klasických spectráckých her se dočtete v Tečce, kde je také pozvánka na poslední ZLINCON tisíciletí (přijeďte všichni!!!).
Rád bych nyní poděkoval všem spolupracovníkům, dopisovatelům a Spectristům, díky kterým tento časopis vzniknul a nadále se těší velkému úspěchu. Všem přeji do nového roku a tisíciletí (i když to tisíciletí začíná až rokem 2001...) hodně zdraví, peněz, elánu a spectrismu. Díky naší práci a zájmu o správnou věc budeme skutečně moci hrdě říci, že ZXS nejenže přežilo rok 2000, ale zájem o něj nějak výrazně neopadá. Na tomto místě bych rád poděkoval TRITOLOVI, který, ač pod nátlakem jiných povinností, jako světlá vyjímka dostál svích slibů, intenzivně se pustil do tvorby file commanderu pro MB-02+ (MB-Commander), který se v těchto dnech dokončuje a vypadá opravdu SKVĚLE (blíže se o něm dočtete v dalším čísle YS, uvidíte jej na ZLINCONu; těšte se, je na co).
To je ode mne vše, buďte zdrávi, neztrácejte soudnost, zůstaňte s námi a hlavně-věnujte se ZX Spectru-stojí za to! V novém tisíciletí se na stránkách YS na viděnou těší...

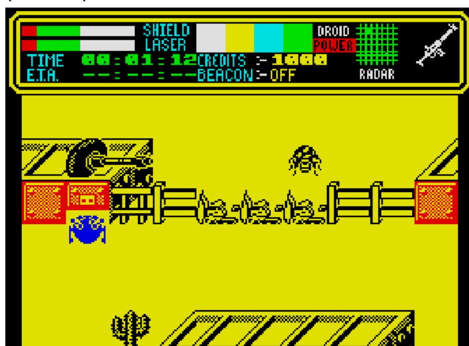
Martin Blažek (BTS), 8BC



*** COLONY ***

Byla jednou jedna planeta a rostly na ní hříbky. Lidi měli tyhle hříbky rádi, protože to byly super drogy. Na té planetě však nebyli jenom houbičky, ale taky pěkně žravé přerostlé hmyzí potvory, které žrali všechno, jak houbičky, tak i lidičky a když byl hlad tak nepohrdli ani kusem plotu. Tak tam naši feťáci nechali postavit kolonii na pěstování houbiček. Postavili obrovské sklady a taky plantáže houbiček. Aby je nikdo nesežral tak tam poslali dálkově ovládaného robota. A právě vy máte možnost ovládat tento spásný stroj.

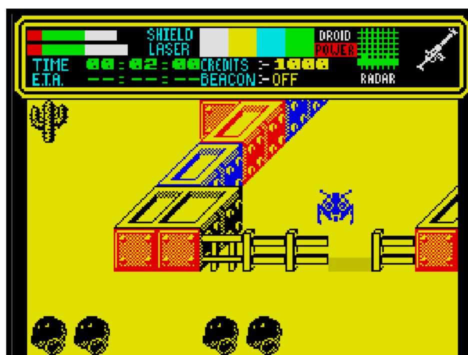
Vaším jediným a nekonečným úkolem je pěstovat houbičky a bojovat s hladovým hmyzem. Za každou vypěstovanou houbu dostanete jistý peníz, za který nakupujete další semínka, ploty, pasti, apod.



Aby se vaše šance na přežití zlepšily, je váš robot vyzbrojen laserem a máte k ruce jednoho automatického droida do kterého musíte dát baterku a on potom běhá po kolonii a střílí jako by si dal 10 dávek houbiček.

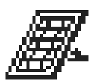
Hlavní úkoly pro udržení kolonie v chodu.

Aby vás mohli zásobovat, musíte si objednat náklad, který vám dopraví raketou. Poté co si něco objednáte, tak se vám objeví čas přistání rakety a vy můžete jít po svých starostech. Těsně před přistání však musíte zapnout naváděcí paprsek, který loď navede na přistání. Potom si musíte věci o odnosit od přistávací plochy do skladů, aby je nikdo nesežral. Pokud nezapnete paprsek tak loď rozháže náklad po celé kolonii i kolem ní. Po přistání musíte paprsek vypnout, protože hrozně žere elektriku.Dostatek elektriky je důležitý pro funkci paprsku, pastí u vchodu a dobíjení vašeho laseru a štítu.
Elektriku získáváte ze solárních panelů, ale ty se dají sežrat tak je doplňujte se zásob. Elektrika se postupně dobíjí do stanice a pokud klesne pod 30 jednotek tak přestanou fungovat pasti u vchodu.

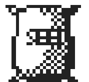


V colony je velice pěkná grafika

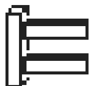
Své plantáže můžete chránit pomoci plotů. Máte k dispozici tři druhy-ocelový, dřevěný a ostnatý drát.Ale i ploty jsou k jídlu. Nebo můžete pokládat pasti, které jsou na jedno použití. Pak si můžete koupit lepší laser, štít a dokonce i vlastní život. Droid, který stojí u přistávací plochy se dá nastartovat pokud na něj položíte baterku.Pak začne pobíhat a vraždit. Ukazatel jeho energie je nahoře na panelu. Pokud mu baterky dojdou tak se zastaví a musíte dát další. Myslím, že je nezničitelný.
Colony se nedá dohrát, je to jak v Simcity. Takže si užijte farmaření a při přistání rakety nelezte na přistávací dráhu.



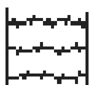
Solární panely - dodávájí elektriku pro doplnování energie štítu, laseru a paprsku. Pokud není, je nutno postavit víc panelů.



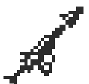
Semínka - v každém pytli jich je 10 používají se na sadbu hřibů. Dají se položit jenom na plantáže.



Dřevěný plot - plot ze dřeva moc nevydží, ale dobré něco než nic.



Ostnatý drát - tento plot je dobrý jako fast food pro hmyz-dvakrát hryzne a je po plotu.



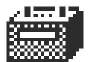
Mega laser - Lepší verze vašeho laseru, zbytečně vyhozené prachy.



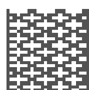
Náhradní život - pokud máte strach o svůj život, tak si kupte tento náhradní.



Automatický robot - tento stroj začne po vložení baterie běhat všude možně a vraždit na potkání.



Baterie - baterka do robota, stačí ji položit na něj a je to.



Ocelový plot - nejlepší plot. Pokud to jde dávejte pouze ocolové ploty. Pokud je kus sežraný můžete ho nechat opravit ve skladu kde jste ho vzali.To platí i u ostatních věcí.



Past - stačí položit a pokud na ni hmyz šlápne tak ho to zabije. Past je na jedno použití.



Štít - lepší verze štítu

-STRY-

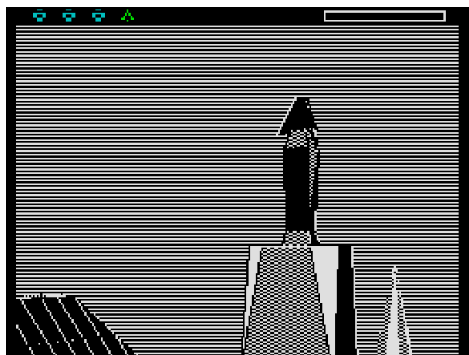
Nápad:	
Hratelnost:	
Grafika:	
Zvuk:	
Vedikt:	Příklad pro naše zemědělce

THE SENTINEL



Na spectru je spousta her s fantastickým nápadem a perfektním provedením, ale The sentinel vede na celé čáře.

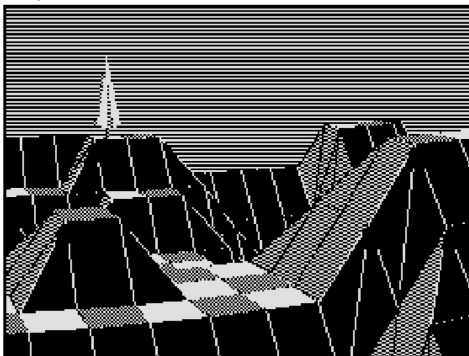
Celá hra se odehrává ve fantastickém 3D prostředí, které je generováno podle vstupního kódu. Na Spectru je to jedna s prvních her s 3D grafikou, která je dělaná stínováním a ne drátama. Ze svého robota se můžete rozhlížet na všechny strany, přesouvat se v prostoru, no proste dokonalý 3D prostor. Sentinel má 10000 krajin, o které můžete bojovat, takže vás jen tak neomrzí.



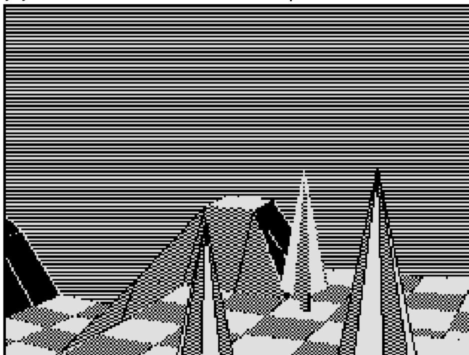
Sentinel se majestátně rozhlíží nad svým územím

V každé krajině se nachází Sentinel, vládce nad krajinou, který se otáčí a hledá vetřelce, pokud nějakého objeví, tak z něj začne vysávat energii a měnit ji na stromy. Vy, ale máte stejné schopnosti jako on, můžete absorbovat stromy, bloky, nebo roboty na energii a naopak. Má to ale jedno omezení, nemůžete pohltit věc, která je výš, než vidíte. Můžete však vytvářet bloky, ty

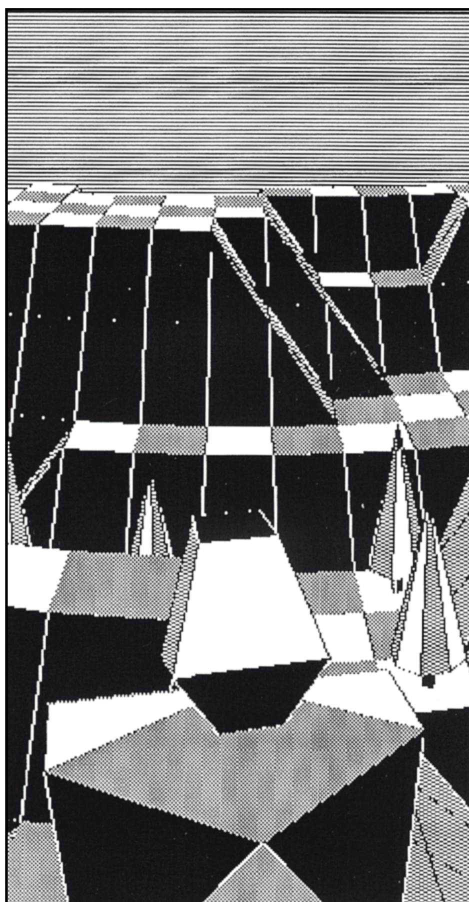
stavět na sebe a potom na ně umístit robota, do kterého se přenesete. Tím se dostanete výš a můžete pohltit víc stromů a svého starého robota a tak se postupně dostat výš než Sentinel a pak ho pohltit a tak tím získat tuto krajinu.



Sentinel však není žádný sebevrah a tak se během toho co se dostáváte nahoru snaží každého vetřelce, kterého spatří, pohltit a vy se pak musíte rychle přenést někam kde vás zrovna nevidí. Pokud vás začne Sentinel nebo jeho pomocníci vysávat a vy potřebujete rychle zmizet můžete zmáčknout "H" a budete teleportováni na náhodné místo v krajině. Sebere to ale určité množství energie. Jestliže ji nemáte dost tak se rozpadnete na prach a to je váš konec. Pokud však pohltíte Sentinela, tak se přesunte na jeho místo, pokochejte se pohledem na krajinu a potom zmačkněte "H" a telepotujete se do další krajiny, přitom obdržíte její číslo a heslo a můžete pokračovat dál. V



dalších kolech se můžete setkat s více Sentinely v jedné krajine a těm můžou pomáhat ještě další hlídky - Sentry (všechny musíte zlikvidovat). Slále však platí, že kdo je nejvýš bere všechno.



Zvukově je Sentinel na super úrovni a nápadem nemá konkurenci.Já osobně by zařadil The Sentinel na vrchol tvorby na Spectru.Kvalitní hra na kvalitním počítači.

Ovládání:

U - Otočit o 180˚	A - Pohltit objekt
H - Nouzový teleport	T - Vytvoř strom
Z - Zvuk zap/vyp	B - Vytvoř blok
C,V, -Barvy	R - Vytvoř robota
P - Pauza	Q - Transport do dalšího robota

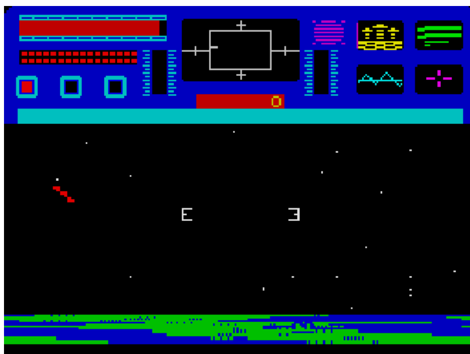
- STRY -

Nápad:	■■■■■■■■■■■
Hratelnost:	■■■■■■■■■■■
Grafika:	■■■■■■■■■■■
Zvuk:	■■■■■■■■■■■
Vedikt:	Vrchol všeho

Star Raiders II

Máte-li chuť si zahrát na spasitele lidstva, tak máte dobrou příležitost. V této výborné střílečce se stáváte posledním přeživším bojovníkem po invazi nepřátelských ufonů.

Máte k dispozici raketu vybavenou lasery, raketami a bombami. Lasery ničíte nepřátelské stíhače, raketami létající talíře a bombami se ničí nepřátelská města. Vaše loď má taky štíty a energii, kterou doplňujete na třech stanicích v naší sluneční soustavě. Tyto stanice však mohou být zničeny nepřáteli, tak si je dobře hlídejte.



Začínáme na zemi

Začínáte na zemi, kde je potřeba zničit pár nepřátelských stíhaček. Pokud máte hotovo, tak stiskem Space vyvoláte mapu soustavy a vyberete si kam poletíte dál. Ve volném vesmíru se pohybují nepřátelské mateřské lodě, které když přiletí na některou s vašich planet, začnou ničit města. Vaším úkolem je města bránit, ale taky ničit města v sousední sluneční soustavě. Ten, kdo zničí všechny města na všech planetách ,vyhrává.
Při ničení stíhaček používáte lasery, ty se však zahřívají a pokud teplota stoupne nad kritickou



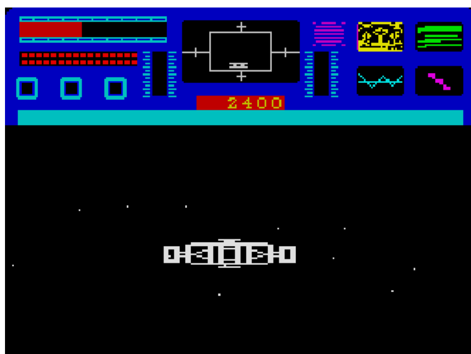
Tento talíř dlouho lítat nebude

mez, tak rychlost střelby klesne a nakonec se lasery zavaří a vy musíte na stanici je opravit. Pokud jsou v místě boje i létající talíře tak, se zaměřovač změní na kříž a přepne se na rakety. Talíře jsou mazané a snaží se uhýbat, ale bojujete-li na planetě tak se snaží ničit i města a v tuto chvíli jsou nejzranitelnější. Hned jak je všechny vystřílíte tak většinou přeletí mateřská loď a začne kolem sebe vytvářet ničí pole, které hodně ubírá energii, ale pár raket ji přivede k efektní explozi. Pokud chcete ničit města tak stiskem "W" přepnete na bomby. Zásoba bomb je však omezena, a musí se doplňovat na staicích. Občas nepřátelská loď přiletí k vaší doplňovací stanici a chce ji zničit.



Nepřátelská soustava

To poznáte tím, že stanice na mapě bliká. Pokud vám zničí všechny stanice tak jste v ... Na ochranu vaší lodi můžete zapnout ochranný štít. Štít však neustále odsává energii.



Mateřská lod'

Po grafické stránce se mi hra velice líbí, hlavně efektníma exploze a rychlostí. Dobré je sladění barev, nenašel jsem žádné zřetelné atributové kolize. Zvuky jsou jenom na speakeru, ale zato na vysoké úrovni.

Popis kláves:

Space - Přepne na mapu
J - Nastavení ovládání (před startem hry)
T - Změna radaru/požkození
W - Změna lasery/Bomby
S - Štít

Nemám nic, co bych této hře vytkl, snad jen to, že je hodně krátká. Jinak je to prostě taková pěkná dobře hratelná střílečka.

-STRY-

Nápad:	■■■■■■■■□□
Hratelnost:	■■■■■■■■■■
Grafika:	■■■■■■■■□□
Zvuk:	■■■■■■■■□□
Vedikt:	Dobrá zábava



AHOOOJ, PŘEJI VŠE NEJLEPŠÍ V NOVÉM TISÍCILETÍ, UVIDÍME SE NA ZLINCONU 2000!!!
-SIR-



Rýchle grafické rutinky na ZX Spektre

O.časť - úvod

Určite ste už všetci videli nejaké to demo, v ktorom všetko na obrazovke veľmi kmitalo, scrollovalo a pohybovalo sa po všeliakých hranatých i oblých dráhach a zdalo sa, ako keby to už proste ani nebol ZX Spektrum s jeho Z80 ale dáky super výkonný počítač s nejakým aspoň 32 bitovým procesorom. A tí z vás, ktorí vedia aspon trošku programovať v strojáku, ste (aspoň v kútiku duše) tiež zatúžili niečo také, alebo dokonca ešte lepšie, naprogramovať.

Cieľom tohto seriálu je ukázať vám nejaké príklady rutiniek, ktoré využívajú procesor Z80 "naplno" a ktoré sa zvyknú používať v rýchlych grafických efektoch v demách (a ktoré tiež tvoria základ mojich dem). V tomto seriáli budú aj rôzne moje finty, ktoré som doteraz ešte nikde inde nezverejnil. Tento seriál som písal hlavne pre ľudí, ktorí už majú určité skúsenosti s programovaním v strojovom kóde a prípadne si už vedia aj sami niečo naprogramovať. Preto nebudem uvedené rutinky vysvetľovať úplne do detailov (ako som to robil v seriáli "Strojový kód pre pokročilých"), miesto toho sa budem zameriavať na hlavnú myšlienku a niektoré najzaujímavejšie aspekty uvedených rutiniek. Z tohto pohľadu by sa tento seriál tiež dal nazdať "Strojový kód pre profesionálov". Najväčším postrachom, nočnou morou z Elm street začínajúceho programátora je samotná videoramka ZX Spektra, lepšie povedané jej šialené usporiadanie (tzv. prekladaná videoramka), ktoré na prvý pohľad nemá logiku. Verte či nie, aj mňa to kedysi strašilo po nociach (aj ja som raz začínal:)) Prvý bodový riadok sa síce začína slušne na začiatku - na adrese #4000, avšak další riadok sa už nenachádza tam kde by sme ho logicky mohli očakávať - začiatok + dĺžka

prvého riadku (#4020), ale na adrese úplne inej. Určite si ešte z dôb kazetových magnetofónov pametáte, ako zaujímavo sa nahrával obrázok. Schválne skúste spustiť toto:

```
CLS: FOR a=16384 TO 23295: POKE a,126: NEXT a
```

Prečo autori ZX Spektra vôbec vymysleli takúto šialenú organizáciu videoramky ? Prečo sa takto zlomyseľne zachovali k poctivým programátorom ? Pozorne si všimnite jednotlivé adresy v pixelovej časti videoramky a aké adresy im zodpovedajú v atribútovej časti. Hneď na prvý pohľad sa ukáže zaujímavá zákonitosť: Nižšie bajty adries všetkých osmych bajtov v pixelovej oblasti a nižší bajt adresy im zodpovedajúceho atribútu sú rovnaké ! (Rada pre tých, ktorí robia v desiatkovej sústave: skúste si tie adresy pozrieť v šestnástkovej sústave - bude vám hneď všetko jasné aj bez výpočtu.) Prečo je to tak ? Operačná pamäť v našom ZX Spektre je realizovaná tzv. dynamickými ramkami. Tieto ramky majú okrem iných jednu špeciálnu vlastnosť:
Keď chceme z takejto ramky prečítať údaj, musíme do nej po zbernici najprv poslať nižší bajt adresy, potom musíme poslať vyšší bajt adresy (alebo najprv vyšší a potom nižší - pre nás to nie je teraz podstatné) a potom nakoniec môžeme prečítať údaj. Bajty adresy si ramka v sebe pametá, preto ak chceme čítať iný údaj, ktorý má povedzme nižši bajt adresy zhodný s predchádzajúcim údajom, stačí do ramky poslať len vyšší bajt adresy. A presne tento princíp využíva ULA v ZX Spektre. Aby sa ušetril čas a procesor bol pri prístupe do videoramky čo najmenej zdržovaný zobrazovaním, je videoramka organizovaná tak, aby ULA mohla načítať bajt z pixelovej oblasti a jeho atribút čo najrýchlejšie.
No dobre, dajme tomu, že už vieme prečo je to tak, ale čo s tým budeme robiť ? Jedna možnosť by tu bola: Existuje rutinka v ROMke ZX Spektra, ktorá nám zo súradníc X,Y priamo vypočíta adresu daného pixela vo videoramke. Táto rutinka sa nachádza na adrese #22B0, pričom na vstupe očakáva X súradnicu v registri C, Y súradnicu v A, a na výstupe vráti v HL adresu bajtu v pixelovej oblasti kde sa nachádza daná pozícia X,Y. Ak by sme použili túto rutinku, vôbec by sme sa už nemuseli starať o umiestnenie bodových riadkov v prekladanej videoramke. Toto riešenie by sme mohli použiť v prípadne, ak by sa tento seriál volal len "Grafické rutinky". Lenže náš seriál sa volá "Rýchle grafické rutinky" a keďže rutinka na #22B0 bude pre niektoré naše požiadavky pomerne pomalá, skúsme nájsť iné riešenie.
Ponúkam vám jednu špeciálnu rutinku, ktorá síce nerobí presne to čo #22B0, ale zato nám tiež čiastočne umožní vyriešiť problém prekladanej videoramky. Rutinku som nazval "dole" - hneď uvidíte prečo. Na vstupe očakáva v HL adresu nejakého bajtu z pixelovej oblasti videoramky a na výstupe vráti v HL adresu bajtu v pixelovej oblasti, ktorý leží hneď pod tým prvým bajtom. Rutinka v podstate realizuje posun "o jeden bod dole".

```
dole    inc h
        ld a,h
        and #07
        ret nz
        ld a,#20
        add a,l
        ld l,a
        ret c
        ld a,#f8
        add a,h
        ld h,a
        ret
```

V ďalších častiach seriálu uvidíte, že aj keď sa to možno na prvý pohľad nezdá, táto rutinka má naozaj veľmi široké (možno aj dlhé a bystrozraké:)) použitie. Napríklad nechýba v žiadnom mojom deme v ktorom sa robí niečo s pixelovou grafikou. Nabudúce si ukážeme, čo to znamená "predpočítaná tabuľka".

- BUSY -

Strojový kód pre pokročilých

lekcia 10

V minulých lekciách sme sa pomerne dlhý čas zaoberali rôznymi matematickými operáciami a prepočtami. Naučili sme sa násobiť, deliť, odmocňovať, ukázali sme si ako treba spracúvať desiatkové a šestnástkové čísla.

Avšak tentoraz celkom zmeníme tému. V dnešnej lekcii (a v niekoľkých nasledujúcich) sa budeme zaoberať rutinkami, ktoré sú veľmi

užitočné a preto v praxi veľmi používané. Rutinky tohto typu sú už štandartnou výbavou všetkých seriózných finálnych programov. Sú to komprimačné a dekomprimačné rutinky. Predstavme si, že máte v pamäti počítača nejaký program. Tento program sa skladá z niekoľkých samostatných častí. Medzi týmito časťami je prázdny priestor - presnejšie povedané - sú tam bajty v ktorých sa nachádzajú samé nuly. Alebo sa v tomto programe nachádzajú nejaké rozsiahle bloky dát, pričom tieto dáta sú "riedke" - to znamená že sa v nich nachádza veľmi veľa takýchto prázdnych oblastí.
Teraz si tento program uložíme na nejaké záznamové médiu (kazeta, disketa, harddisk, streamer...). Tým sme dosiahli to, že časť záznamového média si pamätá iba tie nuly. Teraz si predstavme, že na médium namiesto úseku nulových bajtov uložíme iba počet koľko má byť tých núl a nejakú informáciu, ktorou zabezpečíme aby sme neskôr vedeli tento počet identifikovať (odlíšiť od samotných dát programu). Tým pádom dosiahneme to, že na priestore pamäťového média, kde sa nám predtým zmestili iba dva programy, sa nám teraz zmestia napr. tri také programy. No nie je to perfektné ? Táto činnosť sa odborne nazýva komprimácia (alebo tiež pakovanie).
Keďže komprimovať priamo pri ukladaní programu na pamäťové médium je značne náročné a procesor by to už nemusel stihnúť, zvykne sa to v praxi robiť tak, že sa tento program skomprimuje najprv v pamäti a takto vopred skomprimovaný program sa uloží na pamäťové médium. Skúsme teda vymyslieť rutinku, ktorá vezme nejaký úsek pamäti, skomprimuje ho a takto skomprimovaný ho uloží zase na iné miesto v pamäti.
Keď sa nad tým trošku zamyslíme, tak prídeme na to, že ono je to vlasne činnosť podobná ako robí inštrukcia LDIR, samozrejme len s tým rozdielom, že LDIR nekomprimuje. Zaujímavé je na tom práve to, že naša rutinka potrebuje na vstupe presne tie isté informácie ako už spomínaný LDIR - t.j. adresu a dĺžku daného úseku v pamäti a adresu, kam má uložiť tento úsek skoprimovaný. Ak naša rutina potrebuje na vstupe tie isté informácie ako LDIR tak si pre jednoduchosť zvolme, že jej tie informácie budeme dávať v tých istých registroch ako pre LDIR - t.j. HL bude adresa daného úseku, v BC jeho dĺžka a v DE bude adresa miesta, kam sa má uložiť skomprimovaný úsek pamäti. Týmto pádom si vytvoríme akýsi inteligentný LDIR, ktorý sa navonok od normálneho bude líšiť iba tým, že bude komprimovať. No a teraz poďme k samotnej komprimácii. Ako informáciu pomocou ktorej sa identifikuje počet je v našom prípade najvhodnejšie použiť priamo ten bajt, ktorého sa týka tento počet. Čiže bude to nula. Pri komprimácii teda budeme nenulové bajty kopírovať ako klasický LDIR, ale ako náhle naďabíme na nejaké nuly, do výsledného skomprimovaného úseku uložíme iba jednu nulu (to je tá informácia identifikujúca počet) a potom počet koľko núl sa to vlastne nachádzalo v pôvodnom úseku. Počet si pre jednoduchosť zvolme jednobajtový. To nám dáva možnosť vyjadriť hodnoty z intervalu 0-255. Keďže však hodnotu 0 nebudeme potrebovať (predsa nebudeme do výsledného úseku ukladať informáciu že v pôvodnom bolo nula núl!) tak si povedzme že hodnota 0 bude znamenať počet 256 núl.
Ale čo ak sa náhodnou v úseku pamäti vyskytne za sebou viac ako 256 núl a jednobajtové počítadlo nám nebude stačiť ? Aj tento prípad musíme ošetriť. Zvykne sa to robiť tak, že po 256. nule sa napíše do skomprimovaného bloku záznam o tom, že núl bolo 256 a nuly v pôvodnom úseku sa začnú zase počítať odznovu. Teda ak bolo v pôvodnom úseku napr. 258 núl, do skoprimovaného úseku sa zapíšu tieto štyri bajty: 0,256,0,2.
No a po týchto teoretických základoch môžeme priamo prikročiť k samotnej rutinke. Na začiatku rutinky sú ako ukážkový príklad do registrov HL,DE,BC vložené hodnoty ktoré predstavujú pixlovú časť videoramky (práve tu sa zvykne vyskytovať veľmi veľa núl) na počítači ZX Spektrum. Teda s týmito ukážkovými hodnotami táto rutinka skomprimuje obsah videopamete a uloží ho na adresu #8000.

```
pack    ld hl,#4000    ;príklad adresy
                       ;pôvodného úseku
        ld de,#8000    ;príklad adresy
                       ;voľného miesta
        ld bc,#1800    ;príklad dĺzky
                       ;pôvodného úseku
komlop  ld a,(hl)      ;vezme bajt z
                       ;pôvodného úseku
        ldi            ;skopíruje ho do
                       ;nového úseku
        or a           ;je to nula ?
        jr z,kompck    ;ak áno tak odchod
                       ;na spočítanie núl
```

```
komend  ld a,b          ;inak - spracovali sme
                        ;uz celý úsek ?
        or c            ;kontrola nuly v BC
        jr nz,komlop    ;ak nie tak spracovanie
                        ;ďalšieho bajtu
        ret             ;ak áno tak
                        ;návrat z rutinky
kompck  ld xh,a         ;inicializácia počítadla núl
komnul  inc xh          ;zväčšenie počítadla po
                        ;jednej nule
        jr z,komset     ;ak nám počítadlo
                        ;pretieklo tak koniec
        ld a,b          ;ak sme náhodou
        or c            ;narazili na koniec
                        ;pôvodného úseku
        jr z,komset     ;tak tiez koniec
        ld a,(hl)       ;je tu ešte jedna nula ?
        inc hl          ;korekcia ukazovateľov
        dec bc          ;na pôvodný úsek
        or a            ;ak tu ešte je jedna nula
        jr z,komnul     ;tak skok v slučke
                        ;počítania núl
        dec hl          ;inak vrátenie
                        ;ukazovateľov
        inc bc          ;lebo sme uz došli na
                        ;nenulový bajt
komset  ld a,xh         ;koniec počítania núl
        ld (de),a       ;zápis počtu núl
        inc de          ;a následné zvýsenie
                        ;ukazovateľa
        jr komend       ;skok na test či sme uz
                        ;spracovali celý
                        ;pôvodný úsek
```

Na domácu úlohu skúste porozmýšľať, ako by sa dal z takto skomprimovaného úseku získať znovu ten náš pôvodný úsek pamäti.

- BUSY -

Vyšší programovací jazyky

Být spectristа v podstatě znamená být taky assemblerista - bez toho se dnes prostě nedá existovat. Ale ne na všech úlohách se chce člověk vysilovat psaním rutin, když by stačil jeden příkaz dejme tomu Pascalu s vhodnými parametry.

Běžně je rozšířena představa, že milovníci vyšších jazyků si na Spectru na své nepřijdou, opak je naštěstí pravdou, a dlužno podotknout, že v některých ohledech na tom jsme lépe než na PeCi.

BASIC

O Basicu se dnes tvrdí, že ho vynalezl student Bill Gates a prodal ho jakési firmě. Je to lež jako věž a Billova propaganda. Bill i MS-DOS musel ukrást jistému polákovi M.Z. (tato dvě písmena tvoří tuším hlavičku .EXE souborů), a proč ho pojmenoval Disk Operating System, když diskety nepodporoval a pracoval jen s kazetou, je dodnes záhada. My ale víme, že Basic vyvinuly Bell Laboratories jako multiuživatelský multitaskingový systém (k jednomu počítači se připojilo několik terminálů a několik lidí najednou na nich provozovalo svoje Basicovské úlohy). Spectrum tohle umí taky - vzhledem k přítomnosti jediného televizního výstupu a jediného klávesnicového vstupu nás nesmí překvapit, že zvládne obsloužit celého jednoho uživatele (vtip).

Basic má ZX naštěstí zabudován rovnou jako operační systém (svého času byly i počítače, které jako OS měly zabudován Forth nebo dokonce Pascal), ku jeho zdokonalení přicházejí Mega Basic, skvělý Beta Basic (ve verzi 4.0 naprosto nepřekonatelný), které ovšem neumí kompilovat, Laser Genius Basic a jeho kompiler, interpretující i kompilující, s pro někoho příjemnými rozšířeními.

Kromě interpretů a různých rozšíření se objevily i kompilátory toho Basicu, který je vestavěný, většinou s různými omezeními. Některé nepřekládají do strojáku, ale jenom do tzv. P-kódu, který potom interpretují, jiné zase potřebné runtimy nesou samy v sobě a přeložený kód se na ně v CALLech odvolává - kompilátor nebo aspoň jeho část musí být v paměti (Tobos, Colt).

Překladače pochopitelně nepřekládají úplně všechno, mají nějaká omezení, většinou je to neschopnost práce s desetinnou čárkou. Mistrem v tomhle je IUC, který je přitom skutečně jeden z nejlepších a nejrychlejších compilerů. Práce s ním opravdu připomíná psaní assemblerovských programů pomocí příkazů Basicu - něco jako když pracujete s makry. A schopnost vkládat rovnou strojákový program do překládaného kódu je také k nezaplacení, i když ho musíte ťukat v číslech.

Spectrum je ale počítač neomezených možností a tudíž nemají spectristé omezení rádi. Proto vznikl HiSoft Basic, který je skutečně perfektní, skvěle překládá, omezení není příliš mnoho a nabízí i definice typů

proměnných a podobně. Dokonce od něj vznikla i verze pro 128čku, kterou ale považuji za děsný šmejd. Má problém s příkazem PLAY, nemá žádná jiná rozšíření, nepočítám-li nepovedené menu. Kromě této 128verze ale existuje i 48kový Hi Basic Plus, upravený snad jugoslávskými crackery, který má vychytané některé chyby a navíc má zabudovánu řadu užitečných drobností, například pro předávání obsahu basicových proměnných do strojáku a podobně. Stále však platí některá omezení, například maximálně dvourozměrná pole.

Plně kompatibilní optimalizující kompilátor, u kterého podle manuálu jediný příkaz, který nefunguje, je CONTINUE, se nazývá BLAST.

Ten nejnovější existuje ve dvou verzích. 3.7 je plná verze, 4.0 nebo SUPERBLAST+ obsahuje IBLAST, který používá jen celá čísla (zato běží ještě o trochu rychleji).

Co je na BLASTu zajímavé, je to, že žere děsně moc paměti, kompilace MEMORY TO MEMORY vám vezme jen asi dvoukilový program. Proto jako vstupní a výstupní zařízení můžete používat i tape a mrkvodrajv (pochopitelně v jakékoliv kombinaci - s výjimkou TAPE TO MICRODRIVE). Je to dáno tím, že kompilátor je opravdu mohutný, zato výsledný kód stojí za to, je rychlý (40x proti originálu) a neomezený v možnostech. Například proměnné ukládá stejně jako BASIC, takže strojákovou rutinou si je můžete měnit či vypisovat stejně jako u BASICu, dokonce fungují i rozšíření error rutin - při kompilaci syntakticky podivného příkazu - třeba LOAD* na D40 - se sice zobrazí varování, ale takovýto "pomrvený" příkaz se uloží tak jak je a při jeho vykonávání se emuluje Basic - takže lze používat i povely pro takové periferie, jako je D40 !!!

Opravdu - kompilátor, který vám "sežere" D40kový LOAD*, to tu fakt nebylo…

I u Blastu si můžete vybrat, jestli chcete P-kód nebo stroják, ale podle mne asi P-kód v životě nepoužijete.

Protože při kompilace TAPE TO TAPE se musí kompilovaný program nahrávat po částech, existuje i TOOLKIT, který vám program příslušným způsobem upraví. Asi by nebylo od věci se STRUČNĚ zmínit o ovládání.

TOOLKIT:

Příkazy začínají hvězdičkou, číselné parametry (x,y,z) se oddělují čárkami. Lze pracovat i s bloky řádek (x-y je x až y, -y je od začátku až do y, x- je od x do konce, tečka je aktuální řádek, bez uvedení rozsahu se zpracuje celý program).

*Ex - EDIT řádky x

*Cx,y - COPY řádku (nebo bloku) x na řádek y

*Dx - DELETE řádku či bloku x

*Mx,y - přesun řádku nebo bloku x na pozici y (původní pozice se vymaže)

*R<rozsah>,x,y - RENUMBER, nové číslování začne od x s krokem y (defaultně 10).

*F<rozsah>,string - FIND textu (moc užitečné), při hledání povelu Basicu napište nejdřív THEN, pak příkaz a pak THEN smažte

*S<rozsah>,string1,string2 - SEARCH AND REPLACE, najde a nahradí

*T - trasování, vypisuje čísla prováděných řádků, SPACE sníží rychlost, SHIFT 0 (nebo delete???) trasování zastaví. *T zároveň zapíná sledovací režim, který lze vypnout pomocí *U

*R - KILL, smaže všechny REM řádky, ve kterých nejsou příkazy pro kompilátor BLASTu

*W<rozsah>,filename - uloží blok jako saostatný program v BASICU

*V - VARS, zobrazuje stavy různých proměnných systému

*L - LIST, vylistuje proměnné Basicu

*Jx - JOIN, spojí řádek s dalším

*A/*G - zapíná a vypíná čekání na stisk klávesy při hledacích funkcích (*F, *S)

*Q - quit

*Bfilename - BLAST SAVE, uloží program tak, aby ho BLAST uměl načíst při kompilaci z pásku

BLAST:

Některé verze jsou chráněny heslem, ale existuje i cracknutá verze (Blast Bez Sifri).

Defaultně je nastaven překlad do P-kódu. Změnit to můžete příkazem REM!PCODE nebo REM!MACHINECODE. Strojový kód a P-kód se mohou v programu kombinovat, ale nesmí na sebe skákat příkazy goto/gosub (před každým takovým příkazem je potřeba přepnout typ kódu), takovou blbost asi v životě ale nepoužijete. Dalším REM příkazem, který musí být na prvním řádku, je REM!AUTORUN, který způsobí, že se zkompilovaný program po nahrání spustí (BLAST sejvuje i BASICový loader). Zajímavou možností je napsat příkaz, který normální Basic neprovede, ale kompilátor ho zkompiluje. Takový příkaz se píše do řádku začínajícího REM%, například REM%: GOTO 1000.

A příkazy:

NEW - vymaže BLAST

*Q - totéž
*N - NEW bez snazání BLASTu
*I - nastavení vstupního zařízení (stiskni R,T,M pro RAM, TAPE, MICRODRIVE)
*O - nastavení výstupního zařízení
*C - COMPILE z nastaveného vstupního na nastavené výstupní zařízení
*R - umožní spustit program skompilovaný do RAM, při kompilaci do RAM ale nelze volat strojákové podprogramy.

To by bylo ve STRUČNOSTI k BLASTu vše, kdo chce vědět víc, ať si taky něco zjistí sám...

Jen taková poznámka: na MBčku mi kompilace TAPE TO TAPE nepracuje.

PASCAL

I tady existují interprety, jako třeba český Mikrobáze Pascal. Uznajte ale sami, že interpret Pascalu je úplně k ničemu. Naštěstí hodní kluci od HiSoftu stvořili i kompilátory Pascalu, a to velmi schopné. Populární je hlavně řada HP4, její poslední verze je TM1.6, ve které je vychytána naprostá většina chyb verzí předchozích (HP4D, HP4S, HP4T, HP4TM1.4, HP4TM1.5) a přibyly i některé nové funkce, dokonce i možnost INCLUDE jiného zdrojového textu.

Kompilátor je úsporný, rychlý, s nepatrnými omezeními, umí INLINE strojového kódu (viz IUC, taky v číslech), má dobře vyřešenu práci s kazetou, nahrává i oblasti proměnných, takže uložíte třeba nějaké seznamy, uchovávané v paměti dejme tomu jako record, pochopitelně si velmi dobře rozumí s MBčkem, veškeré runtimy nemají víc jak 3,5 kila. Nepoužívá sice proměnné typu FILE, ale je to vyváženo mnoha jinými zlepšeními, které na rozdíl od jiných Pascalů umožňují vzít počítač pohodlně do ruky. Možná existují i verze, které podporují jiné diskové systémy než kazetu, já jsem ale nic takového nepotkal.

Editor je, jak je u HiSoftu zvykem, úplně příšerný. Asi vám přijde nezvyklý, ale upozorňuji vás, že svého času byly takhle debilně ovládané řádkové editory standardem, užívaným hlavně pod CP/M. Protože HiSoft psal překladače pro různé Z-80kové počítače tak, že jádro bylo společné a lišil se jen interfacing (čtení klávesnice, tisk znaku), je logické, že použil "univerzální" způsob editace, který funguje všude (a programátoři na něj byli druhdy zvyklí). Takže odporné editory GENSu, Pascalu, Cčka jsou osmibitovým průmyslovým standardem a nic s tím nenaděláme.

Neee, to není tak docela pravda, viz GENS s vylepšenými editory. I u HP4TM1.6 můžete vestavěný editor nahradit svým oblíbeným, ale příručku, na kterou se manuál odkazuje, kde se píše, jak to udělat, jsem nepotkal (Alteration Guide). Nezbývá mi než si myslet, že to bude asi podobný způsob jako u GENSu.

Druhou možností je použít editor externí a příkaz pro import textu z cizího editoru. I to Pascal umí.

Navíc vzhledem k tom, že jádro kompileru, ale i runtimů je hardwarově nezávislé, všechno se totiž volá pomocí jakýchsi vektorů, umožňuje ona Alteration Guide kromě připojení vlastního editoru i takové věci, jako zahrnout do runtimů obsluhu jakéhokoliv zařízení pro vstup či výstup...

Nebýt editoru, byl by to nejlepší Packal, jaký jsem kdy viděl. A spousta bývalých spectristů si ho děsně libuje, zvlášť při srovnání s PeCí.

Víc se o něm šířit nebudu, manuál, který jsem poskládal z originálního manuálu prvotních verzí a dodatků k opravám verzí následujících, má skoro 50 stránek. (http://fly.to/gama)

Od té samé firmy existuje i HP 80, viděl jsem ho ale jen z rychlíku a teď ho nemůžu nikde sehnat, ani na Internetu. Má ale editor se 64 znaky na řádek, tím se zvyšuje přehlednost, a snad nemá ten úchylný řádkový systém editace (to ale nemůžu tvrdit jistě).

C

Ano, i tenhle jazyk na Spectru najdete. Zásluhu na tom má zase HiSoft se svým C 1.0 a novějším C 1.1.

A ten, kdo Céčku rozumí, je jimi vskutku nadšen (to se mne netýká, ale nadšen jsem taky).

Tohle Céčko není jen kompilátor, připomíná trochu Basic ZX Spectra, kromě psaní programu v editoru se můžete přepnout do módu, kdy napsaný příkaz nejde do zdrojového textu, ale předá se rovnou kompilátoru, přeloží, a sérii takhle zkompilovaných povelů můžete rovnou nechat vykonat (narozdíl od Basicu i opakovaně).

Opět se setkáváme s debilním řádkovým editorem a la CP/M, pokud ale máte jakýkoliv editor, který řádky ukončuje kódem 10, můžete ho použít. Kód 13 se v Céčku ignoruje.

Kompilátor je obohacen proti jiným Cčkům o příkazy specifické pro Spectrum, nemá implementován pouze FLOAT (ale snad by šlo napsat si pro něj knihovnu?). Nejzajímavěji je udělaná

práce se soubory - je totiž podobná jako na UNIXu.
Cčko ukládá soubory jako sekvenční. Základní verze umí pracovat s mikrodrajvem a tejpem. Zařízení rozpoznává tak, že tejp je definován pouze jménem souboru, u mikrodrajvu je i jeho číslo (1:DATA je soubor mikrodrajvový, DATA je soubor na kazetě). Mikrodrajvové soubory mají v hlavičce uvedenu délku nula, tak se nelekněte!
Nejvychytanější jsou sekvenční soubory na kazetě. Nejprve je uložena hlavička, zbytek je v 514 bajtů dlouhých sektorech (bloky 2 bajty identifikace, 512 bajtů data). Při vstupu se sektor načte do bafru, z něj se data berou, po uprázdnění se nahrává další sektor, stejně se postupuje při výstupu dat, která se ukládají do bafru a když ten se zaplní, uloží se jeden sektor.
Viděl jsem i verzi 1.1 pro BETASHiT, upravili si ji asi rusové, sekvenční soubory jsou k nerozeznání od obyčejného souboru CODE. Mám ji sice schovanou (s krátkým manuálem v ruštině), ale nic bližšího k ní asi nenapíšu.
Vzhledem k tomu, že v Céčku programovat neumím a manuál je velmi dlouhý, sežeňte si informace u někoho kompetentnějšího. Rozhodně to ale vypadá tak, že je to zas jednou schopný kompilátor schopného jazyka, díky tomu, že proti normálnímu Cčku není skoro vůbec omezen, má šanci stát se oblíbeným (je to mohutný nástroj, který Spectristé asi nedocenili). Podle ohlasů ze zahraničí je to daleko povedenější věc než nějaký HiSoft

Pascal...

Protože tohle je poslední překladač od HiSoftu, připomeňme si, že HiSoft stále existuje a píše překladače pro Amigu, například Pascal...
A když už jsme u těch rusů, RST-7 ze skupiny Code Busters, jinak autor assembleru TASM, dokončil práce na svém vlastním překladači Cčka, ještě jsem ho sice neviděl, ale dá se čekat, že to nebude žádné ořezávátko, nicméně nemajitelé BETASHiTu si asi neškrtnou.

FORTH

Zásobníkový jazyk, který byl vytvořen pro řízení hvězdářských teleskopů v reálném čase. V jeho potrhlých zkratkách se zřejmě vyznal jen sám autor, nicméně já sám pamatuji dobu, kdy programátoři všechny solidnější aplikace, jako řízení strojů v mlékárnách, psali právě v tomto jazyku. Není tedy divu, že existuje i pro Spectrum, celkem povedenou mutaci má na svědomí Psion, i když compiler to nebude. Spectrista ale asi nepotřebuje hýbat s teleskopem, a tak je mu tento didakticky prý velmi vhodný program k ničemu.
Psion taky ještě nezanikl, pro jejich kapesní počítače Series 5 dokonce existuje emulátor Spectra!

FORTRAN

Z mně neznámých důvodů se používal hlavně v socialistickém zdravotnictví, a to proto, že se tehdy jelo hlavně na socialistických compech. Spectrum ovšem socialistický comp není, takže Fortran pro něj se shání těžko.
Výjimkou byla Klinika zobrazovacích metod v Motole, kde se tehdy jelo tvrdě na Spektráčích, a ještě roku 1996 byly skříně Ústavu biofyziky 2.LF UK plné asi padesáti Didaktiků. Dnes už ale vyměkli a zobrazují na PeCi.

PROLOG

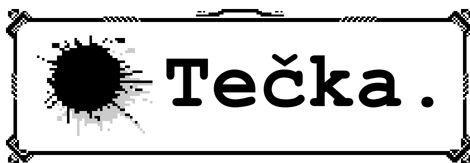
To je skvělý kříženec inteligentní databáze s databázově orientovaným jazykem, jeho okleštěním vzniklo slavné SQL. na Spectru existuje jeho populární mutace Micro Prolog, ke které za žádnou cenu nemůžu sehnat manuál (hledal jsem na Internetu, a vypadá to, že technické knihovny mají k dispozici jen pár exemplářů, které ještě nevyhodili, neklame-li mne ale paměť, kazetu originálku i s manuálem jsem viděl svého času u Honzy Hanouska...).
Micro Prolog je jeden z nejslavnějších produktů firmy LPA, která rovněž nezanikla a věnuje se nadále Prologu a databázím, i když na PeCi a spol.
I když ho neumím ovládat, nastíním vám jeho možnosti.
U obyčejné databáze máte pevně dané položky se vztahem dvou prvků. Třeba Jméno:Matsoft Ulice:Lotyšská 8. Stejně musíte strukturovat i ostatní záznamy o jiných lidech. a teď si vzpomenete, že Matsoft je vydavatel ZX Magazínu (skrytá reklama na jiný spectristický časopis). Na to, abyste to v databázi uvedli, musíte vytvořit novou položku - třeba Poznámka, a doní si to poznamenat, nebo položku Vydavatel ZXM a do ní Ano a u všech ostatních záznamů do kolonky Vydavatel ZXM napíšete Ne...
Než se dostaneme k tomu, jak to dělá Prolog, si představte jinou aplikaci, třeba nemoci, jejich příznaky a léčení (to už nejsou jen tak data, ale

blíží se to už pojmu ZNALOSTI tak, jak je vymezuje umělá inteligence). U většiny chorob máte bolesti břicha, u jiné většiny bolesti hlavy, u některé nevíte, jak vzniká, ale víte, jak léčit, u jiné znáte lék, ale vznik je vám utajen, další přichází v pěti různých formách, jednu způsobuje jedna baktérie, jinou jiná, a my chceme z databáze HNED zjistit, jaké tato bakterie metabolizuje cukry... Je to prostě chaos, jaký panuje ve všech přírodních vědách kromě matematiky a fyziky. Neůžeme si dát přesnou strukturu a do ní doplnit data, naopak máme haldu dat a s přidáváním dalších dat neustále strukturu formujeme, případně i radikálně měníme (změna klasifikace leukémií nebo jiné nehody).
Prolog totiž neprovazuje dva pojmy, ale tři: Entita, vlastnost (nebo vztah) a hodnota [ENT(PROP:VAL)]. Tedy ne Jméno:Matsoft, ale Máma-má-mísu, Lev-žere-maso... Navíc umožňuje dědit vlastnosti a podobně.
Dejme tomu:
Tučňák (je,pták)
Netopýr (má,křídla)
Pták (má,křídla)
Zeptejte se nyní Prologa, jestli má tučňák křídla. Odpoví Ano.
Všimněte si, že do vzniklé databáze klidně přidám Tučňák (výskyt,Antarktida), objevila se v ní informace zdánlivě nesouvisející s dosavadní strukturou, ale Prolog to nezmátlo a já nemusím vůbec definovat nové položky týkající se místa žití daného živočicha. A klidně přidám i Netopýr (výskyt,Evropa), Netopýr (výskyt,Amerika) a Netopýr (výskyt,Asie), pořád je všechno v pohodě...

Expertní systémy

Když dáte informacím číselnou váhu, třeba výskyt v procentech, můžu pak vyhledávat taky na zadanou přesnost shody (sčítání dvou neostře ohraničených množin, tomu se říká FUZZY logika). To má zaseobrovský význam hlavně v biologických systémech, které se nechovají úplně podle defiice.
Vezměme si nemoc nebo bakterii. S nějakou pravděpodobností se u ní vyskytuje nějaký znak (50% glukóza+, 100% galaktóza+, 70% laktóza-), já jsem vypěstoval něco, co chci určit (bacouch Glc-, Gal+, Lac-) a nechám si ho vyhledat v databázi. Takový v ní sice není, ale je tu určité procento shody s jiným evidovaným druhem, který se liší pozitivitou glukózy, ale s nízkou váhou, takže to na 73,3 procent může být jedinec tohoto druhu.
A s nemocí podobně.
Třeba slepák: 70% bolest břicha, 15% bolest hlavy, 50% zvracení, 20% průjem.
Tyfus: 60% břicha, 80% hlavy, 40% zvracení, 30% průjem.
Proženu touto databází pacienta, který nemá bolest břicha, bolí ho hlava, zvrací a nemá průjem. Systém mi nemoci seřadí od nejpravděpodobnější do nejméně pravděpodobné (tyfus 57,5%, slepák 43,7%).
Jak vidíte, od Prologu k expertnímu systému není zas tak daleko, myslím si, že pro programátora je prolog daleko lepší databází než nějaký blbý Masterfile či Datalog 2 Turbo. A to jsem mluvil pořád jen o jeho databázové stránce, ne o programovacím jazyku a schopnosti řešit úlohy (ideální je na hledání nejkratší cesty a jiné učebnicové příklady).
A neříkejte mi, že ta procenta u příznaků chorob jsou ve skutečnosti jiná. Já to vím, byl to jen příklad.
A tím jsme skoro opustili půdu programovacích jazyků a dostali se k umělé inteligenci, takže bude lepší to skončit.

- +GAMA -



V poslední Tečce tohoto tisíciletí se dočtete o kolorovaných spectrákých klasikách a dozvíte se podrobnosti o ZLINCONu 2000. Pěkné počteníčko.

ZLINCON 2000 - Povinnost pro všechny!

8BC si dovoluje všechny Spectristy pozvat na poslední spectristickou akci tohoto měsíce, roku, dekády, století, tisíciletí...

ZLINCON 2000!!!

Letošní ZLINCON 2000 se koná na stejném místě (dům dětí a mládeže Astra v Příluku u Zlína), ve stejné době (víkend před Vánocemi, tj. 18-19/12/99), čili jako každý jiný rok. Organizátory akce je jako obvykle 8BC (tj. LMN

[063/277256]+BLS [0603/543256] +ZRŮDA [0602/575817]).
Co uvidíte? Nejnovější MB-Commander (super-rychlý file manager pro MB-02+) od Trotola, ULA-Pro (grafický modul pro ZXS), prezentaci operačního systému pro harddisk a CD-ROM na ZX Spectru...
Co s sebou: Spektráč, přezůvky, spacák, Kč 128,- (účastné) a nějaké kapesné.
Na letošní ZLINCON 2000 musí přijet všichni!!! Vstoupíme společně se Spektráčem do nového tisíciletí!!!!!



Pokud jste nabyli dojmu, že vám chceme povídat o nějakém nové hardwarové periferii (nebo snad novém Spektráči), musím uvést vše na pravou míru-jste totiž na obrovském omylu.
Spec256 je nový emulátor ZX Spectra pro PC... cože?! Že už máte emulátorů, že nevíte co s nimi? Tak tento určitě nemáte, je totiž jiný.
Spec256 je zvláštní emulátor podporující speciální stejnojmennou technologii, pomocí které je možno mít klasické spectrácké obrázky obarveny 256tibarevnou paletou. Když pak hrajete hry, které tuto technologii podporují, máte pocit, že paříte nejnovější hit na Playstationu (nebo Nintendu 64). Samozřejmě, že takovégo vylepšení přináší určitou daň:
-takto kolorované hry můžete hrát pouze na PC a pod emulárotem Spec256
-v současnosti je k dispozici poute 12 titulů využívající Spec256 (Phantis, Jet Pac, Game Over 1 & 2, Solomon´s Key, Abu Simbel, Army Moves 1 & 2, Cybernoid, Sabre Wulf, Knight Lore,Underwurlde).
Na druhou stranu tyhle hry vypadají opravdu moc hezky, bohužel to není dostatečně vidět na přetištěných obrázcích (proto, máte-li přístup k Ingernetu, navštivte **http://www.emulatron-ia.com/emusdaqui/spec256/uso-eng.htm**)
Celý nápad Spec256 vzniknul někde ve Španělsku, když si několi Spectristů po zhlédnutí PC verze Manic Minera řeklo, že by stálo za to převléct do nového hávu i jiné spectrácké klasiky. Přepisovat celý kód hry se jim ovšem nechtělo a proto použili originální strojový kód, ke kterému vytvořili nové grafické reprezentace veškeré původní grafiky. Napsali emulátor Spec256 a vše bylo na světě.
Na celé věci vás určitě upoutá skutečnost, že původní SNA/Z80 soubor zůstal nedotčen a je tedy spustitelný ve kterémkoliv jiném emulátoru. Celý nápad je postaven na myšlence, že klasický procesor Z80 disponuje osmibitovými registry, zatímco při emulaci na PC jsou dispozici 64-bitové registry (totéž platí o mapě paměti). Pokaždé, když je emulována instrukce procesoru Z80, je interpretována tak, jakoby pracovala s grafickými daty (tento "paralelní" procesor autor pojmenoval Z80_GFX). Z80_GFX pracuje s rozšířenou paměťovou mapou a výsledky jsou fantastické. Samozřejmě, že veškerá nově vzniklá (=kolorovaná) grafika musela být manuálně nakreslena na PC.
Co říci na závěr? Hry v Spec256 vypadají opravdu bombasticky, ale na hratelnosti her to nic nemění. Hry v Spec256 na svém ZX Spectru sice nerozchodíte, ale truchlit nemusíte. Originály stejně vypadají nejlíp.
Pozn.: na výše zmíněné internetové stránce můžete najít jak samotný emulárot Spec256, tak i dodatečnou grafiku k uvedeným hrám.

-BLS-



>HAVE FUN WITH YOUR SPECTRUM IN Y2K!